Amendments to the Claims:

The following listing of claims will replace all prior versions, and listings, of claims in the application:

1. (Currently Amended) A code translation device that translates a source code of a source processor into a target code of a target processor, the device comprising:

a memory; and

a controller, the controller dividing the source code into ~~translated~~ code blocks ~~of the target code~~ based on a target processor register capability, wherein the controller identifies source register types as data registers or address registers of the source processor and corresponding target registers of the target processor that correspond to each of the source register types, the controller selecting one or more selected source register types and one or more maximum numbers of corresponding target registers that correspond to the selected source register types as the target register capability.

2. (Original) The device of claim 1, further comprising a branch detector, the branch detector identifying one or more instructions of the source code that either includes a branch, a loop return or an entry point for a branch or loop return.

3. (Original) The device of claim 2, wherein the controller generates one or more source code blocks based on the identified instructions, each of the source code blocks beginning immediately after an identified instruction and includes all consecutive instructions following the identified instruction up to an instruction immediately before a next identified instruction.

4. (Canceled)

5. (Original) The device of claim 4, further comprising a register detector, the register detector detecting a number of source registers that are used and/or updated in one or more instructions of each of the source code blocks.

6. (Original) The device of claim 5, wherein the controller generates one or more translated code blocks for each of the source code blocks based on a number of selected source registers detected by the register detector and the maximum numbers of corresponding target registers.

7. (Original) The device of claim 6, further comprising a stub generator, the stub generator generating a head stub and a tail stub for each of the translated code blocks.

8. (Original) The device of claim 7, wherein a head stub associated with a translated code block initializes one or more target registers used by the associated translated code block, the target registers being initialized by retrieving register values from a source register map that stores values of the source registers during execution of the translated code blocks.

9. (Original) The device of claim 7, wherein a tail stub associated with a translated code block saves values of one or more target registers used by the associated translated code block in a source register map after execution of the translated code block.

10. (Original) The device of claim 9, wherein the source register map includes storage space for one or more values for each source register accounting for instruction execution delays, the tail stub saves values of the target registers in one or more appropriate locations in the source register map to account for the instruction execution delays.

11. (Currently Amended) A method for translating a source code of a source processor into a target code of a target processor, the method comprising:

identifying a target processor register capability; ~~and~~

dividing the source code into ~~translated~~ code blocks ~~of the target code~~ based on the target processor register capability;

identifying source register types as data registers or address registers of the source processor and corresponding target registers of the target processor that correspond to each of the source register types; and

selecting one or more selected source register types and one or more maximum numbers of corresponding target registers that correspond to the selected source register types as the target register capability.

12. (Original) The method of claim 11, further comprising identifying one or more instructions of the source code that include a branch, a loop return or an entry point for a branch or loop return.

13. (Original) The method of claim 12, further comprising generating one or more source code blocks based on the identified instructions, each of the source code blocks beginning immediately after an identified instruction and includes all consecutive instructions following the identified instruction up to an instruction immediately before a next identified instruction.

14. (Canceled)

15. (Original) The method of claim 14, further comprising detecting a number of source registers that are used and/or updated in one or more instructions of each of the source code blocks.

16. (Original) The method of claim 15, further comprising generating one or more translated code blocks for each of the source code blocks based on a number of selected source registers and the maximum numbers of corresponding target registers.

17. (Original) The method of claim 16, further comprising generating a head stub and a tail stub for each of the translated code blocks.

18. (Original) The method of claim 17, wherein a head stub associated with a translated code block initializes one or more target registers used by the associated translated

code block, the target registers being initialized by retrieving register values from a source register map that stores values of the source registers during execution of the translated code blocks.

19. (Original) The method of claim 17, wherein a tail stub associated with a translated code block saves values of one or more target registers used by the associated translated code block in a source register map after execution of the translated code block.

20. (Original) The method of claim 19, wherein the source register map includes storage space for one or more values for each source register accounting for instruction execution delays, the tail stub saves values of the target registers in one or more appropriate locations in the source register map to account for the instruction execution delays.